КІБЕРБЕЗПЕКА: освіта, наука, техніка
CYBERSECURITY: EDUCATION, SCIENCE, TECHNIQUE
№ 2 (26), 2024
ISSN 2663 - 4023
DOI 10.28925/2663-4023.2024.26.702
UDC 004.72:004.77

Maksym Kotov
Master's Degree in Computer and Information Systems Security,
PhD Student at the Department of Cyber Security and Information Protection
Taras Shevchenko National University of Kyiv, Department of
Cybersecurity and Information Protection, Kyiv, Ukraine
ORCID ID: 0000-0003-1153-3198
maksym_kotov@ukr.net

TREE-BASED STATE SHARDING FOR SCALABILITY AND LOAD BALANCING IN MULTICHAIN SYSTEMS

Abstract. Staying abreast with the fast-paced demand surge towards distributed consensus systems has become one of the global trends in the fields of science and engineering. The blockchain technology, its consensus protocols, communication methods, and architectural approaches are prevalent in trustless transactional systems. In that context, one of the key obstacles faced by engineers and interaction peers is the limited scalability capacity entailed with these systems due to consistency and reliability requirements. Modern blockchain systems introduce complexities related to the storage space management, transaction execution latencies, and, in general, the throughput of operations, which stifles the widespread integration of decentralized systems in day-to-day activities. In order to circumvent these limitations, a plethora of inter-chain communication protocols, sharding strategies, and capacity extension methods are being actively developed by both scientific and engineering communities to mitigate initial logical limitations of the proposed consensus technology. Nonetheless, the developed solutions are associated with limitations of their own, often converging to a bottlenecked point in their load balancing approach or sacrificing significantly in finality and latency properties. The purpose of this article is to introduce and describe the tree-based sharding approach of multichain systems. Firstly, this paper describes a general architecture of the proposed network, establishing a foundation upon which the later discussion takes place. Secondly, a set of communication methods involving parents, siblings, and remote branches to exchange transaction data. Lastly, the proposed sharding architecture and its properties are compared with a set of existing strategies towards achieving scalability within the confines of the blockchain technology. Overall, this article presents a novel approach towards building reliable, scalable, and highly efficient multichain systems through a structured tree-like hierarchy of cooperating blockchain networks.

Keywords: blockchain; multichain systems; blockchain state sharding; blockchain load balancing; multichain communication protocols; multichain consistency and consensus; tree-based multichain system.

INTRODUCTION

Centralized coordination solutions are the prevalent method of establishing trusted communication and value exchange platforms. Having a common trusted entity allows multiple unacquainted parties to establish a secure transactional session. The shortcoming is the requirement for an unconditional trust for the centralized entity and its virtue, which, as history shows, sporadically leads to undesired and even harmful outcomes.

As a result, the contemporary demand for a decentralized value exchange solution has been steadily increasing for the last few decades [1]. The foundation upon which these systems are built stems from the blockchain technology developed in the late 20th century. The blockchain in its distilled form could be described as a linked list of hash sums representing an entire interaction history between involved parties [2] – [5].

© M. Kotov, 2024

Initially, the blockchain technology was integrated into a network of coordinating peers, each storing the exact same copy of the chain, also referred to as a ledger. Nowadays, these networks have found ways to expand their flexibility and computational capabilities through the integration of virtual execution environments for their native programs called smart contracts (SCs). This further expanded the horizon of possibilities related to decentralization, paving the way for decentralized exchanges (DEXs) as we know them today [6].

The blockchain network architecture is commonly referred to and abstracted out in a set of hierarchical stratums, viz., the inter-chain communication protocols (the so-called zero layer); the consensus plane (the first layer); scalability and efficiency (the second layer); decentralized applications (the third layer); the web application interface (the fourth layer). This approach allows one to modularize the constituent components of the network, addressing each limitation and challenge in an independent way [7], [8].

The key constraints associated with the current state of decentralized systems are related to the latency, throughput, and the ever-growing storage capacity demands for keeping the entire interaction history saved on every node. To address them, myriads of approaches and extensions are being actively developed on different blockchain architecture levels by both scientific and engineering communities.

Since the initial cardinality of connected clients to the blockchain network was not high, the described limitations did not have an immediate and evident impact on the interaction experience. Though, the situation has been dynamically shifting with a growing influx of new network peers. One of the first remedies for the clients with limited hardware capabilities was to allow both full and limited history mode storage on a node in addition to the ability to interact in two distinct modes: spectator and validator [9]. This approach allows for ameliorating cost incurs and hardware demands for a subset of users but still requires the network to have nodes that would face the initial limitations.

In turn, layer two solutions aim at providing an additional infrastructure plane on top of the consensus network. Its primary concern is to decrease latency and increase the overall throughput of the network by distributing and offloading computational complexity throughout the network. An example of such technology is rollups that are actively integrated into the modern blockchains to offload program execution and heavy computations from the bottlenecked consensus layer [10], [11]. This approach, nonetheless, does not provide a meaningful solution for scaling the consensus layer network.

One of the recent notions is a shift from blockchain to a multichain paradigm that tries to resolve issues associated with logically limited capacities inside a single network. The multichain concept in its foundation describes a potential way of message passing and consistent asset exchange between multiple blockchain environments. This could be further categorized into sharding and interconnection of unrelated chains [12] – [14].

The purpose of the article is to present a novel multichain architecture, intercommunication, and sharding strategy to circumvent limitations entailed with the original decentralized network design. It is the intention of this article to broaden the spectrum of possibilities related to load balancing and scaling of contemporary decentralized systems.

Throughout this article, theoretical modeling, including both mathematical and graphical representations of the concepts, is leveraged to simplify the integration of the described architecture into the modern networks. As a result, a comparison between the proposed decentralized model and existing multichain networks is provided to streamline the decision-making process made by engineers and architects involved in the development of the said systems.

ISSN 2663 - 4023

TREE-BASED MULTICHAIN SYSTEM ARCHITECTURE

To achieve state sharding and load balancing capabilities, this paper defines a tree-like multichain structure that is comprised of a hierarchical relationship between interconnected blockchains. Its structure includes three key node types called Trunk, Boughs, and Springs; and their amalgamations that will be described later.

The tree-based multichain architecture is shown in the following Fig. 1:

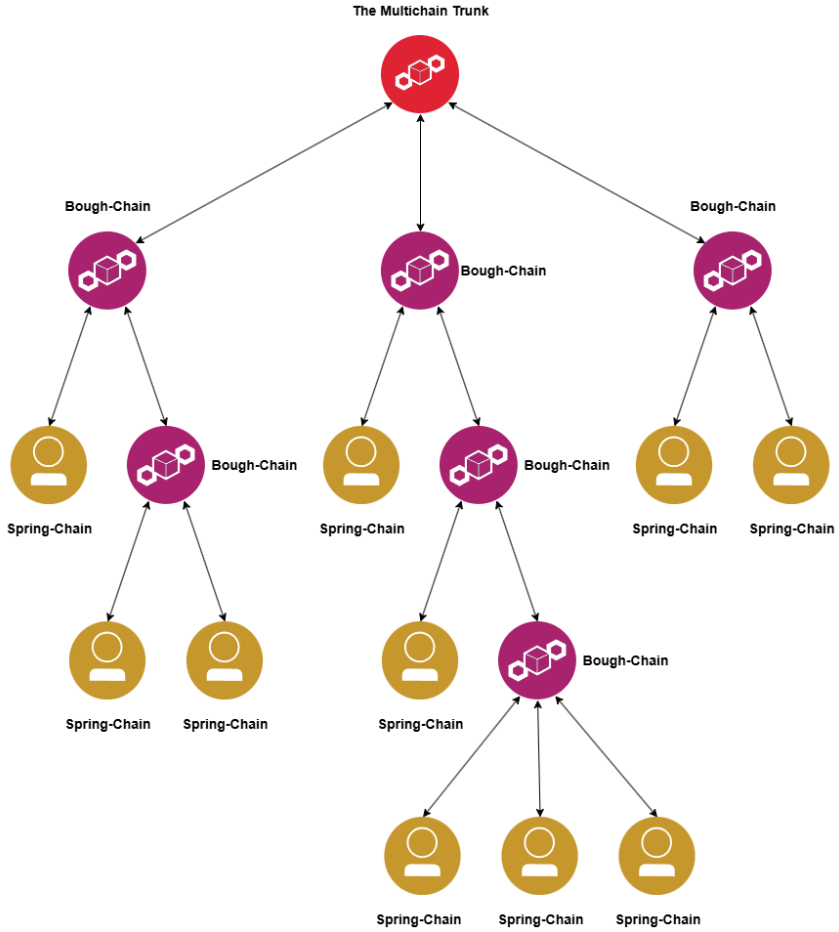

Fig. 1. Tree-based multichain architecture

Let $T = (V, E)$ be a rooted tree representing the interconnected blockchains, where:

- V is the set of nodes (blockchains).
- $E \subseteq V \times V$ is the set of directed edges representing parent-child relationships.
- The root node $r \in V$ represents the Trunk chain.
- Function $\mathcal{V}(v)$ returns a set of validators associated with $v \in V$.

Definitions:

- Parent function: for each $v \in V$, the parent $p(v) \in V \cup \{\emptyset\}$, where $p(r) = \emptyset$.
- Children set: for $v \in V$, the set of children is $C(v) = \{u \in V \mid p(u) = v\}$.
- Bough chains: nodes $v \in V$ with $C(v) \neq \emptyset$.
- Spring chains: leaf nodes $v \in V$ with $C(v) = \emptyset$.

The Trunk is the root node $r \in V$, where $p(r) = \emptyset$, which serves as an initial point for the entire multichain structure. Throughout the entire structure of the multichain system, every

node is allowed to have its own local economy and minted assets, including the Trunk chain. The primary purpose of this chain is to register a set of second-layer Bough chains and coordinate communication between them.

In turn, Bough chains are coordination nodes that serve as local consensus and security management centers for their integrated infrastructure. Bough can also engage with transactions in its own environment and is not restricted only to the management roles. Trunk chain is a special case of Bough chain, being the nexus point of the network. Communication methods established within the scope of Bough chain subnet will be disclosed in the next section.

The Spring chains are the ones that are logically designated to manage users and their interactions as expected from the blockchain ledger in its original design. Spring chains can change their role to Bough when the majority of validators decide that load balancing and sharding capabilities are needed to maintain the operational state.

The amalgamation of multiple consecutive hierarchical Bough chains is called Governance Structure. These structures are defined for and exist in the context of each Bough chain separately, and, by default, Governance Structure ($\mathcal{G}_d$) consists of immediate parent-child relationships:

$$\mathcal{G}_d = \{(v, p(v)) \mid v \in V, p(v) \neq \emptyset\} \quad (1)$$

In turn, a United Governance Structures ($\mathcal{G}_u$) represents a set of Bough chains that have agreed to form an extended Governance Structure.

$$\mathcal{G}_u = \{(v_i, v_j) \mid v_i, v_j \in V_u\} \quad (2)$$

where $V_u \subseteq V$ is the subset of participating blockchains with direct hierarchal dependency that have unanimously achieved agreement about establishing a new $\mathcal{G}_u$.

Chains in $\mathcal{G}_u$ coordinate policies, validator rotations, and inter-chain communications beyond immediate parent-child relationships. If $(v_i, v_j) \in \mathcal{G}_u$ and $(v_j, v_k) \in \mathcal{G}_u$, then $(v_i, v_k) \in \mathcal{G}_u$ holds under united governance agreements.

This architecture allows to achieve load balancing by offload computational complexity related to the management of the subset of minted assets to a set of Spring chains.

First, let's define function $\mathcal{A}(v)$ to denote the set of assets on chain v. Then, for $v \in V$, define $S \subseteq \mathcal{A}(v)$ as the assets selected for offloading. During the process of load balancing on the Brough v, for each asset $a \in S$, the main chain v locks the asset:

$$\text{Lock}_v(a): \quad a \rightarrow \text{LockedState}_v(a) \quad (3)$$

Locked assets cannot be transacted on v until unlocked. equivalents are minted on Spring chain s. The Spring chain s mints equivalent assets a' corresponding to each $a \in S$:

$$\text{Mint}_s(a'): \quad \emptyset \rightarrow a' \quad (4)$$

The locking and minting processes are synchronized in a way that there exists a bijective mapping $\phi: S \rightarrow S'$, where $S' \subseteq \mathcal{A}(s)$. Having said that, the total supply is maintained across chains:

$$\forall a \in S, \quad \text{Supply}_v(a) + \text{Supply}_s(a') = \text{InitialSupply}_v(a) \quad (5)$$

Having established a decentralized multichain system, it is also important to have additional durability guarantee mechanisms. In the case of the proposed model, each validator in $\mathcal{V}(v)$ is required to store historical data of up to n ancestors, where $n \in N$.

For $v \in V$, define $A_N(v) = \{u \in V \mid \text{distance}(v, u) \leq N\}$, where "distance" is the number of edges between v and u in T. This is a target history storage set that must be included within validators $\mathcal{V}(v)$ and could be dynamically extended through the consensus process to guarantee consistency and durability of the network.

ISSN 2663 - 4023

COMMUNICATION WITH PARENT BRANCHES

One of the primary communication channels that has to be established within tree-like multichain architecture is between parent and child nodes. In that context, a communication paradigm could be established in multiple ways, including locking and proof of validity approaches, which we will discuss later.

The communication path between children and parents within tree-based multichain architecture is shown in the following Fig. 2:

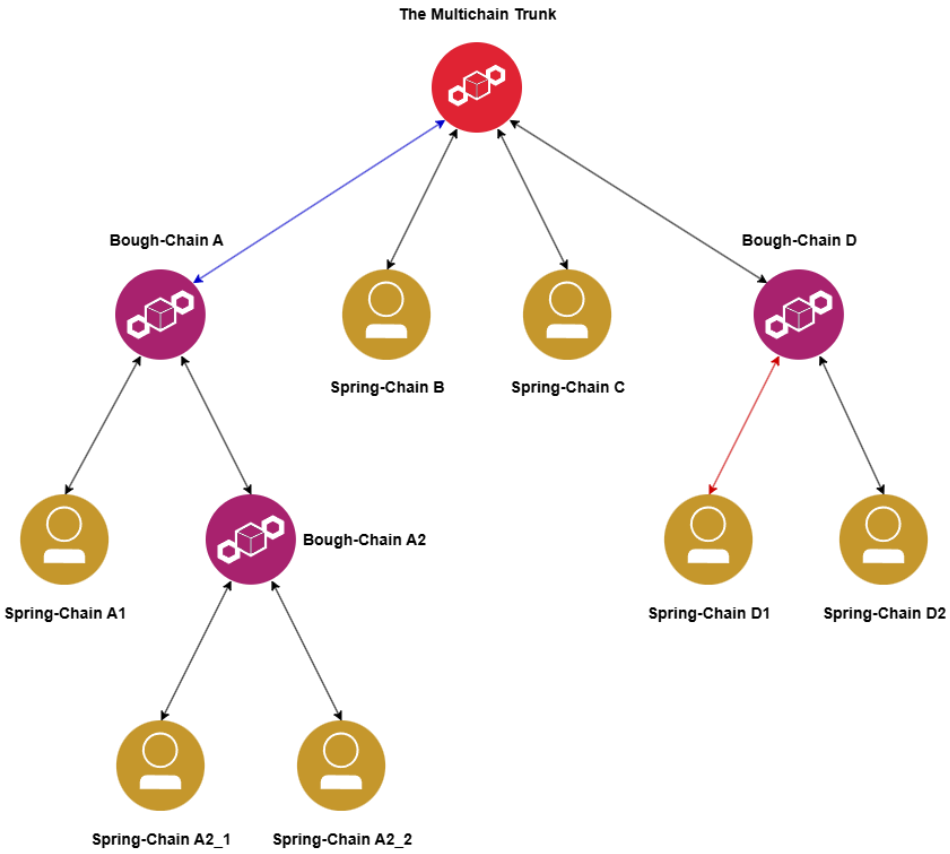

Fig. 2. Communication path between children and parents

Parent-child communication edges are shown between “Bough-Chain A” and “The Multichain Trunk”, also between “Bough-Chain D” and “Spring-Chain D1”. In both cases, the communication would entail the same protocols and properties based on the chosen approach.

The first approach in that case would be resource locking and minting mechanisms. It is characterized by rapid finality, and its security is dependent on the message-passing approach. It is often utilized on Bridges and Wormholes in a centralized manner, reducing the confidence and security of such interactions [15], [16]. When lock-based communication is dependent upon a centralized entity, like a specific service provider represented by a smart contract and its supporting external infrastructure, clients have to fully trust the Bridge provider to correctly transfer assets from one chain to another. The process entails the submission of assets to the representative entity on the holding chain. Subsequently, the entity's off-chain infrastructure detects a new transfer request, initiates a minting operation on the target chain, and transfers the assets to a target account.

The decentralized locking approach relies on the native chain governance and external node inclusion capabilities. The purpose is to have a subset of nodes that would also contain and maintain references to the target chain. That is, a subset of nodes has to be part of both networks, which could introduce complexities but is a more secure and fast way of interaction.

The proof of validity approach will be discussed in the next section and is generally more suitable for controlled sharding. There are multiple approaches to establishing such infrastructure, but since the proposed architecture involves a notion of Governance Structures, that don't necessarily trust each other completely, a locking mechanism is more secure to avoid shard-based attacks.

Consider the security issues as follows: let's say that "Spring-Chain D1" was successfully hijacked by malicious actors. In that case, if the proof of validation is leveraged in communication and asset transfer between parents and children, the Spring could convince its Bough to illegitimately transfer assets as if Bough itself is responsible for such an operation. How this issue is mitigated on the shard level will be discussed in the next section. The point of using locking mechanisms rather than allowing free communication upwards is to isolate the compromised lower layer from having an impact on higher or adjacent networks in the multichain. This is achieved since lock-based communication requires consensus within the parent network and its validation nodes without relying solely on the state transition proofs.

Having said that, let's consider the communication architecture between parents and children within the proposed architecture. Let us define $\mathcal{R}(v) \subseteq \mathcal{V}(v)$ is the set of validators actively participating in the validation process on v.

The following properties apply for each blockchain $v \in V$:

- Subset relationship: $\mathcal{V}(c) \cap \mathcal{V}\big(p(c)\big) \neq \emptyset$. That is, there is a validators subset of c that are also a subset of the parent's validators.
- Roles on parent chain: validators in $\mathcal{V}(c)$ act as observers on $p(c)$ and do not engage in its validation process. Formally, $\mathcal{V}(c) \cap \mathcal{V}\big(p(c)\big) \neq \emptyset$ but $\mathcal{V}(c) \cap \mathcal{R}\big(p(c)\big) = \emptyset$.
- Context access: validators in $\mathcal{V}(c)$ have read access to the state of the parent chain $p(c)$.

When an operation O originates from c, validators in $\mathcal{V}(c)$ create a proposal:

$$\text{Propose: } \mathcal{V}(c) \times O \to M\big(p(c)\big) \tag{6}$$

where $M\big(p(c)\big)$ is the set of messages (proposals) sent to $p(c)$.

The parent chain $p(c)$ processes these proposals using its active validators:

$$\text{Validate: } \mathrm{M(p(c))} \times \mathcal{R}\big(\mathrm{p(c)}\big) \to \{\text{Accepted,Rejected}\} \tag{7}$$

One of the key security advantages in this approach is that the validators on the parent chain have to actually go through the entire transaction validation process that has to comply with the established rules on the chain. In addition to that, locking paths allows one to differentiate between original assets and their wrappers, which in turn could serve as a provenance mechanism that further improves security properties.

An asset $a_c \in S(c)$ is locked on the child chain c, where $S(c)$ represents a set of locked assets on the given chain c: $\mathrm{Lock}(a_c) \in S(c)$. This operation completely restricts access to the locked assets until the corresponding network burns its minted tokens.

A wrapped asset $w(a_c)$ is minted on the parent chain $p(c)$:

$$\mathrm{w(a_c) = Mint(a_c) \in \mathcal{A}\big(p(c)\big)} \tag{8}$$

where $w: S(c) \to W\big(p(c)\big)$, and $W\big(p(c)\big)$ is the set of wrapped assets on $p(c)$.

COMMUNICATION BETWEEN SIBLING BRANCHES

The next communication mode is designed between the sibling nodes in the tree hierarchy. This interaction is contingent upon the proof of validity approach leveraged in numerous contemporary multichain systems. One of its primary aspects is centralized security and consensus management between shards, allowing for establishing a trusted environment for inter-chain asset transfer since the security model is shared between networks [17] – [20].

The communication path between siblings within tree-based multichain architecture is shown in the following Fig. 3:

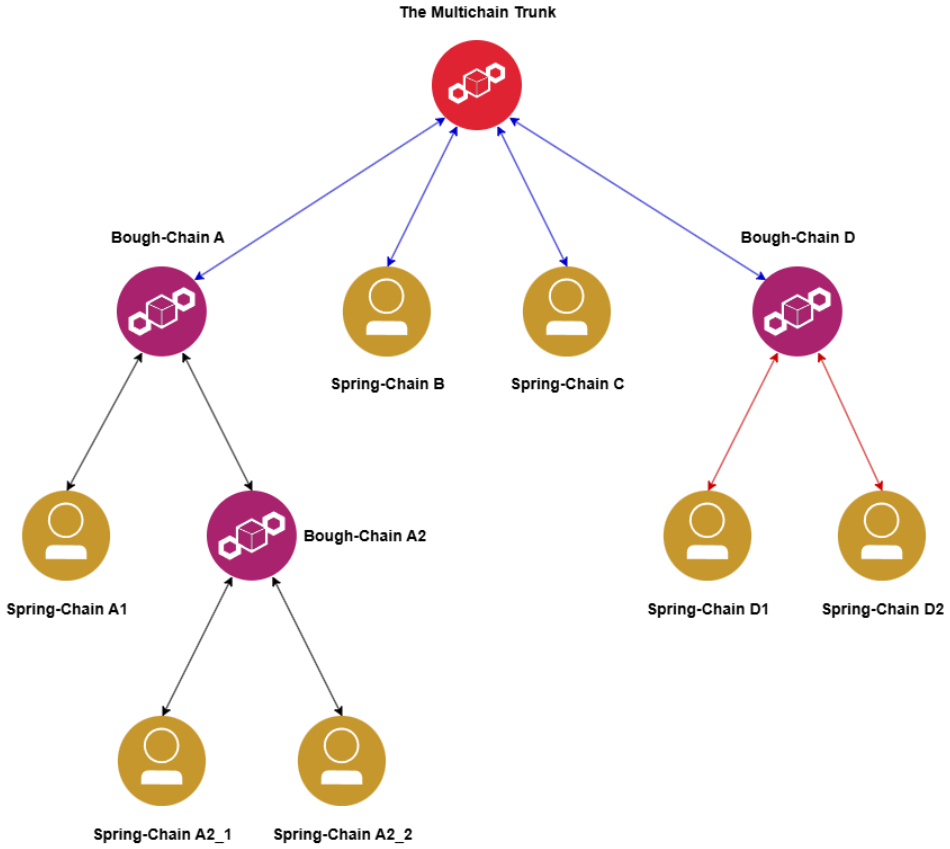

Fig. 3. Communication channels between siblings

In the context of the proposed multichain model, the groups of blockchain networks that are being managed by a central security and coordination authority are called the Governance Structures discussed in the previous section. In that context, Fig. 3 shows 2 Governance Structures united under the “Multichain Trunk” and “Bough-Chain D”.

It becomes apparent that Bough nodes act as children and parents in the proposed model. That is also a reason why asset transfer upwards needs to go through the lock and release process in addition to the on-level validation and consensus to mitigate security risks that might entail with such an architecture. The only Bough that is not managed by any other parent is the “Multichain Trunk”, being a single nonconformant entity in the set. That does not mean, however, that the root cannot be changed. Multiple independently launched networks that conform with the protocols described within the scope of this architecture could be united by adding another higher-level network, becoming a new “Multichain Trunk”.

The backbone of such interactions is a proof of validity method. In its essence, it relies on compliant behavior of every shard with minimalistic state transition proofs based on Merkle trees being shared with one another through the centralized coordination network [17] – [20]. In that context, let S be the global state space, which could be also described as a large set of key-value pairs scattered through the sharded amalgamation of decentralized networks. Each shard's state at time t is a finite subset $S_t \subseteq S$. We define a cryptographic hash function $H: \{0,1\}^* \to \{0,1\}^n$, assumed to be collision-resistant and one-way.

Let's represent S_t as an ordered sequence of leaves $(x_1, x_2, \dots, x_m)$, each encoding a piece of state data. The Merkle tree structure is generated by recursively hashing sets of direct child nodes for the entire set of state values $x_m \in S_t$:

$$\mathrm{R_t} = \mathrm{H}\big(\mathrm{H}(\mathrm{x_1}) \parallel \mathrm{H}(\mathrm{x_2}) \parallel \cdots \parallel \mathrm{H}(\mathrm{x_m})\big) \quad (9)$$

In practice, it is most common to see the Merkel tree as a binary structure, and each non-leaf node is computed as:

$$N_{\text{parent}} = H\big(N_{\text{left}} \parallel N_{\text{right}}\big) \quad (10)$$

The process is repeated in a hierarchical order, until we reach the single root R_t, representing a snapshot of the entire local state.

A state transition from S_t to S_{t+1} is defined by a set of transactions $\mathcal{T}$ that deterministically map:

$$\mathrm{S_{t+1}} = \Gamma(\mathrm{S_t}, \mathcal{T}) \quad (11)$$

Here Γ is a deterministic clean function, that, by definition, must result with the same outcomes given the same parameters. The corresponding root changes from R_t could be represented for some Merkle root computation function f as:

$$\mathrm{R_{t+1}} = \mathrm{MerkleRoot}(\mathrm{S_{t+1}}) = \mathrm{f}(\mathrm{H}, \mathrm{S_{t+1}}) \quad (12)$$

Shards (e.g. Springs in the context of the proposed architecture), rely on the centralized network and state roots submitted to it to verify asset transfers from other sibling networks. To achieve that, every shard $i \in \{1, \dots, K\}$ periodically constructs and send shard roots $R_t^{(i)}$ to the corresponding "Bough Chain".

Let $\{R_t^{(1)}, \dots, R_t^{(K)}\}$ be the set of shard roots at time t. These are combined into a global commitment B_t:

$$\mathrm{B_t} = \mathrm{g}\left(\mathrm{H}, \{\mathrm{R_t^{(i)}}\}_{\mathrm{i=1}}^{\mathrm{K}}\right) \quad (13)$$

where g is a predefined aggregation function responsible for generation of a single cryptographic commitment to all shard states. This approach allows to minimize the amount of overhead related to the maintenance and scaling of shards.

When cross-shard transaction happens, in order to verify a particular state element $x_j \in S_t$, a validator obtains a Merkle proof, a minimal subset of tree nodes $\mathcal{P}$ which are sufficient to recreate R_t:

$$\mathrm{R_t} = \mathrm{RecomputeMerkleRoot}\big(\mathrm{H}, \mathrm{x_j}, \mathcal{P}\big) \quad (14)$$

If R_t matches the expected root recorded within B_t, then x_j is verified as part of the correct state. Given the one-way and collision-resistant properties of H, any tampering would alter R_t, thus, the verification process would detect inconsistency and abort operation.

Beyond that, the proposed architecture includes a notion of a United Governance Structures ($\mathcal{G}_u$) which represents a set of Bough chains that have agreed to form an extended Governance Structure. The operational complexity does not differ significantly from what we've described for a single Governance Structure and relies on the same principles with the addition of the higher level of global commitments and validator assignment coordination.

COMMUNICATION BETWEEN REMOTE BRANCHES

The final communication approach that could be established within the confines of the proposed multichain model is interaction between distant branches. The transactions could be executed in multiple ways, both of which rely on the locking mechanisms due to security and efficiency reasons. The first approach is applicable between multiple consequently connected networks by leveraging parent-child relationships and performing multi-chain hops. The other allows establishing an arbitrary communication link between distant nodes but requires preliminary agreements to be established, as will be shown later.

The communication path between remote nodes within tree-based multichain architecture is shown in the following Fig. 4:

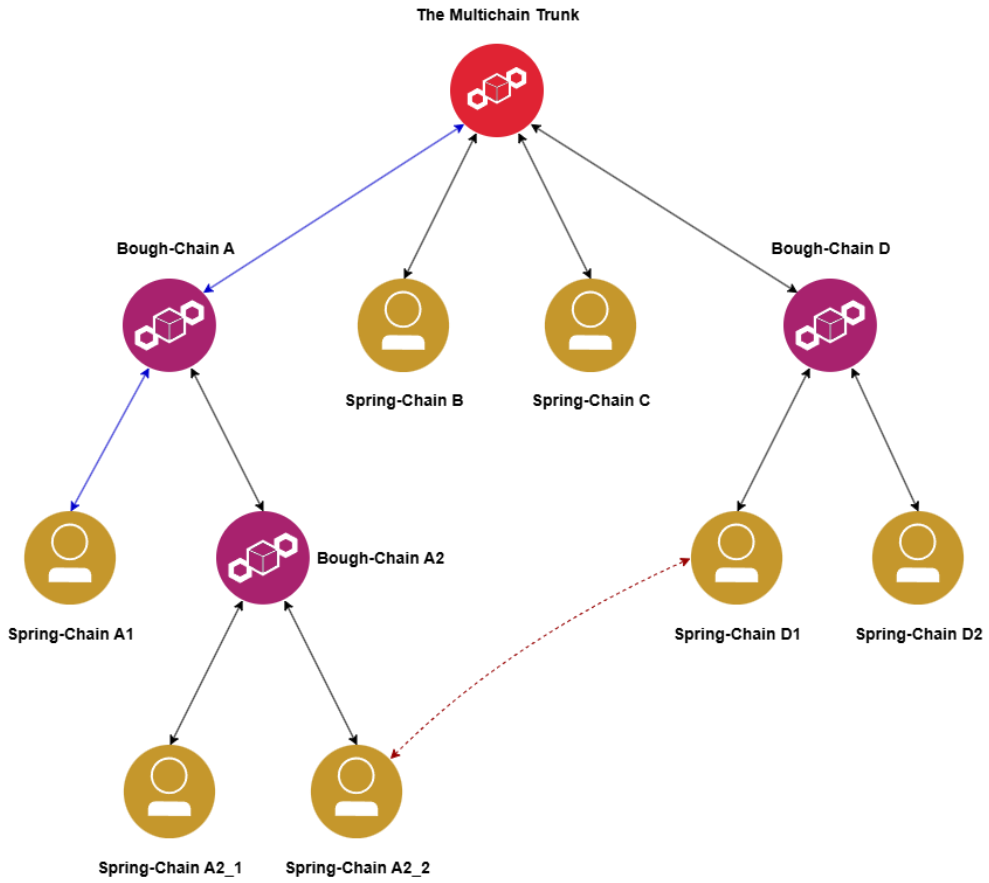

Fig. 4. Communication channels between remote nodes

As shown in Fig. 4, “Spring-Chain A1” is communicating through the “Bough-Chain A” to the “Multichain Trunk” by performing a multichain hop. Such an approach could be characterized with a relatively fast finality since in the proposed architecture, parent and children networks are overlapping, which allows for rapid communication and finality. In addition to that, multichain hops leverage heightened security characteristics, since the cross-chain transaction would be validated on multiple layers before being applied to the target network. In that context, communication can be established both in direction to the root and from it, as the final step, communication between the siblings could be initiated in complex scenarios.

Though such an approach is associated with a potentially considerable overhead, since transactions must be recorded on every intermediary blockchain. This could potentially lead to congestion, but it's intrinsic in the design of the system that highly cohesive blockchains should be placed relatively close to leverage infrastructure's capabilities. In such a case, remote message passing would not lead to congestion or inefficiencies since the farther networks are the less likely they will have to frequently initiate transactions between themselves.

In that context, let's consider now a set of blockchains V organized into a rooted tree structure $T = (V, E)$, where each node $v \in V$ represents a chain and each edge $(u, v) \in E$ represents a direct parent-child link.

For any two chains $i, j \in V$, let $P_{ij} = (v_1 = i, v_2, \dots, v_k = j)$ be the unique simple path connecting i to j in the tree T. Each edge $(v_m, v_{m+1}) \in E$ along P_{ij} has an associated transaction fee $f_{v_m, v_{m+1}} > 0$. These fees model the cost of transmitting a single message through the intermediate chain corresponding to that edge. In that case, the total cost of a single message communicated via multi-hop from chain i to j is the sum of all edge fees along the path:

$$\mathrm{C}_{\mathrm{ij}}^{(\text{multi-hop})} = \sum_{(\mathrm{v_m}, \mathrm{v_{m+1}}) \in \mathrm{P_{ij}}} \mathrm{f}_{\mathrm{v_m}, \mathrm{v_{m+1}}} \tag{15}$$

Moving on to the governed connections between remote nodes, suppose a direct governance link is established between two non-adjacent chains i and j. This link is formed by selecting a subset of validation nodes $\mathcal{S}_{ij} \subseteq \mathcal{V}(i) \cap \mathcal{V}(j)$ that will act as a jointly trusted bridging committee.

The creation of such a governance link incurs a connection overhead cost for the network $O_{ij} > 0$, which includes the complexity of governance voting, committee selection, and synchronization among the involved nodes.

Once established, the direct governance link enables communication at a reduced per-message cost $g_{ij} \geq 0$, which is significantly lower than the cumulative cost of multiple hops since the connection is direct and would involve transaction fees in two networks only. Hence, for a given communication session involving N messages between i and j:

$$\min\left\{\mathrm{N} \cdot \mathrm{C}_{\mathrm{ij}}^{(\text{multi-hop})},\ \mathrm{O_{ij}} + \mathrm{N} \cdot \mathrm{g_{ij}}\right\} \tag{16}$$

For a pair of chains (i, j) that need to exchange N messages, the decision to use multi-hop or governance link can be described and reduced to minimizing total communication cost related to transactions propagation:

$$C_{ij}^{(\text{governance})}(N) = O_{ij} + N \cdot g_{ij} \tag{17}$$

If communication is infrequent or if the multi-hop path P_{ij} is short and inexpensive, then the relation could be expressed as:

$$\mathrm{N} \cdot \mathrm{C}_{\mathrm{ij}}^{(\text{multi-hop})} < \mathrm{O_{ij}} + \mathrm{N} \cdot \mathrm{g_{ij}} \tag{18}$$

If communication is voluminous and happens frequently between target chains or if the multi-hop fees accumulate rapidly, a governance-established link becomes cost-effective:

$$\mathrm{O_{ij}} + \mathrm{N} \cdot \mathrm{g_{ij}} < \mathrm{N} \cdot \mathrm{C}_{\mathrm{ij}}^{(\text{multi-hop})} \tag{19}$$

The limitation of such an approach is apparent, since before remote transaction execution could be established, the network has to undergo the governance, negotiations, and infrastructure update steps. To execute an inter-chain transaction, each intermediate network takes fees for transaction processing. Distant multi-hop transfers could not only be inefficient but also costly for the initiators, which incentivizes cohesiveness and, in special cases, the establishment of direct transfer channels between remote branches in the multichain system. Further research should be aimed towards constructing lightweight and secure connections between chains.

COMPARISON WITH EXISTING MULTICHAIN SYSTEMS

The following section presents a comparison between the proposed multichain architecture and a set of the most popular and adopted approaches available nowadays. Starting with Ethereum 2.0, which introduces a sharded approach towards handling decentralized interactions.

The said network relies on an already discussed proof of validation method, with the central control chain called “The Beacon” to manage validators rotation and facilitate inter-shard interactions by storing state roots. The key characteristic of Ethereum’s approach is in the periodic submission of state hashes to the central chain, rather than on demand. This allows to significantly reduce the congestion related to the management of the Beacon chain but also impacts latency and finality of the inter-chain transactions since they rely on submitted state proofs to the Beacon [19].

The sharding architecture developed withing Ethereum and Polkadot is shown in the following Fig. 5:

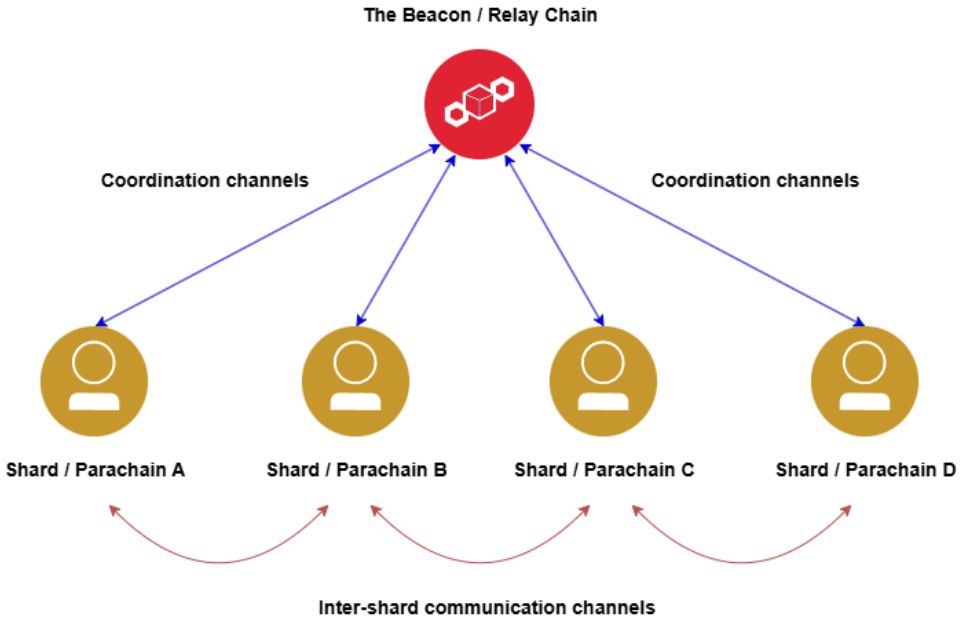

Fig. 5. Ethereum/Polkadot sharding architectures

Polkadot is another example with a similar consensus approach. In its ecosystem, the central chain is called “The Relay”, while shards are called “Parachains”. The shard utilizes Cross-Chain Message Passing (XCMP) protocol for transactions with cryptographic proofs stored on the Relay. The key difference with Ethereum is that the proof is generated for each transaction rather than periodically, which congests the central chain but allows for rapid transaction finality [20], [21].

The difference between these architectures and the proposed one is evident. The proof of validity approach is leveraged between sibling chains, but the communication paths and the structure of the network are not limited by horizontal relations between networks. Instead, the proposed architecture aims to provide a foundation for additional expansions and preserve the security as well as the consistency properties of the communication. The similarities between sibling communications are inspired by Ethereum 2.0 architecture.

ISSN 2663 - 4023

Moving on to the next multichain system, Cosmos presents a technology that aims to unite entirely independent blockchain networks. The system closely resembles a network itself, by providing and establishing direct communication links between decentralized networks and establishing central coordination networks to facilitate interoperability [22], [23].

The multichain architecture developed withing Cosmos network is shown in the following Fig. 6:

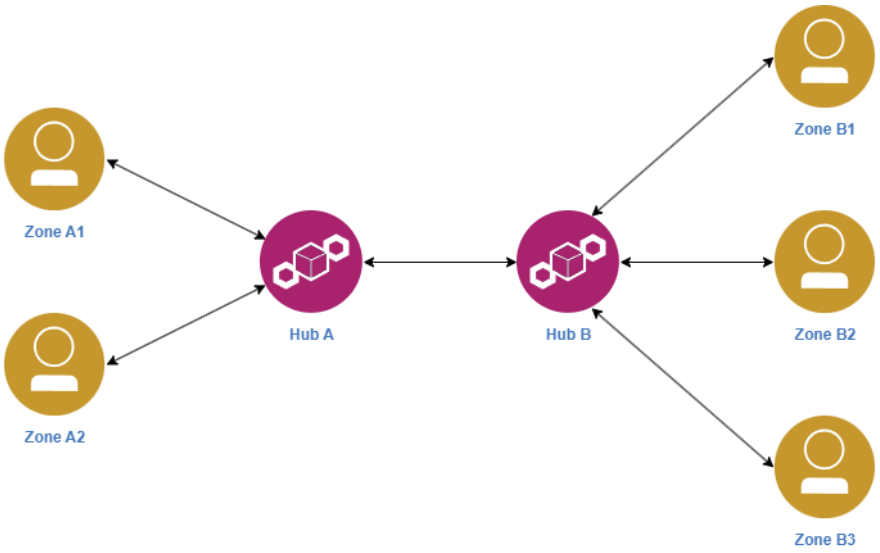

Fig. 6. Cosmos multichain architectures

Cosmos relies on Inter-Blockchain Communication (IBC) protocol to execute inter-chain transactions. The process involves multiple steps and starts with the transaction creation on the source network and its validation by the decentralized network. After that, the IBC relies on external Relays, which could also be controlled by a centralized entity or organization, to construct the package and transfer it to the target chain. The target chain, in turn, must maintain a subnet of Light nodes in its network, which essentially stores a set of public keys and state aggregation from the source network. These nodes receive the packet along with the cryptographic proof to validate the transaction, and if successful, include it in the target ledger. After that, the acknowledge message is sent to the source network through the Relays, to finalize the transaction [22], [23].

This system also introduces Hubs, to avoid redundancy related to establishing and maintaining individual connections between independent networks. Hubs simply serve as a central point that performs transaction forwarding and routing. Nonetheless, Hubs are decentralized networks themselves.

While comparing Cosmos' approach to the proposed architecture, it is evident that the primary purpose of this system is to establish an inter-chain communication medium, rather than coordination, load balancing, or sharding. Cosmos lacks common security orchestration frameworks, which stems from its very purpose to unite independent networks, rather than create a united system.

In addition to that, Cosmos relies on external Relays, that do not impact security but could significantly impact transaction finality and latency. In that regard, this approach could be compared to interconnection systems such as Wormholes and Bridges rather than multichain systems like Ethereum, Polkadot, or the proposed solution.

Lastly, one of the most popular multichain systems is Avalanche, which establishes a subnetting approach in its decentralized architecture. Avalanche is itself a unified system under a single governance and coordination mechanism that incorporates capabilities for multichain management through the division approach [24], [25].

The multichain architecture developed withing Avalanche network is shown in the following Fig. 7:

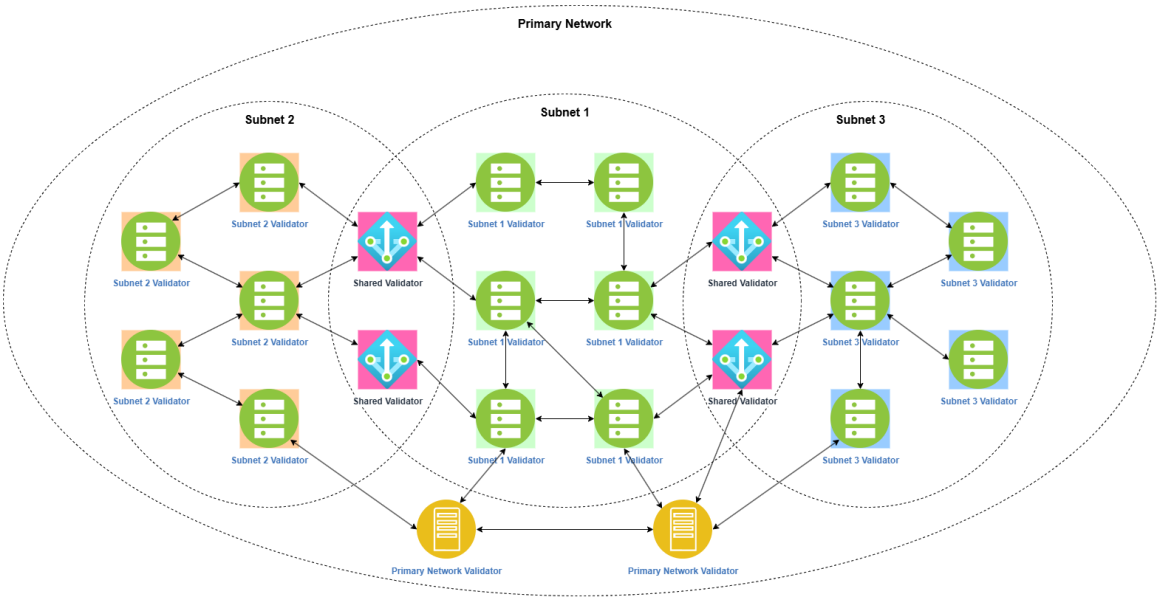

Fig. 7. Avalanche multichain architectures

The primary network is responsible for managing three core chains called C-Chain, P-Chain, and X-Chain to manage all its subdivisions and economy. C-Chain is an implementation of the Virtual Machine, responsible for the execution of Smart Contracts. P-Chain manages the lifecycle of subnets and the coordination of validators that constitute the decentralized network. X-Chain manages the native assets available on Avalanche [24], [25].

As was stated previously, the primary network can register within itself a subnet that can establish its own consensus protocols and infrastructure. In turn, the subnet has to comply with the rules of the primary network to interact with other chains available in the system. The interaction process is built upon Atomic Cross-Chain Transactions (AXC) protocol. It starts with validation of the inter-chain transaction on the source network and propagating it to the destination network, which could be done easily since P-Chain stores the mapping of validators. The festination network verifies the quorum of validator signatures, again, by leveraging the P-Chain and confirms the transaction, after which it could be finalized [24], [25].

When comparing this architecture with the proposed model, it is apparent that Avalanche was designed with interoperability and ecosystem expansion in mind rather than load balancing and sharding. It leverages a strong consistency and security model, since it is shared throughout the network in contrast with the proposed model, which suggests Governance Structures and their unified extensions. Overall, inter-chain transaction finality in Avalanche is rapid, but the security and latency capabilities come at the cost of a single congestion point being the three primary chains that govern the entire network. In contrast, the proposed model does not suffer from a central congested point, since in addition to the Trunk, additional links can be established to join remote blockchain systems.

CONCLUSIONS

The modern requirements towards scaling and load balancing capabilities caused by a rising popularity of decentralized systems inadvertently lead to the surge of research and development initiatives aimed at discovering efficient methods of building multichain systems. As a result, this article provides a detailed description of the novel tree-based state sharding approach for contemporary blockchain systems and networks.

Firstly, this article provides a thorough description of the proposed sharding model, emphasizing the key characteristics, flexibility, and interconnection mechanisms that could be established in the said system. Consideration for durability and intuition behind such an approach are laid down to simplify the decision-making process made by architects and engineers that are working on the development of the new blockchain-based decentralized systems.

Secondly, multiple communication modes are described within this article that stem from the natural properties of a tree-like structure, starting with parent-child branch interactions. Such an interaction is the fastest and the most efficient mode of communication, since child nodes are partially involved in their respective parent networks by design. In turn, communication with siblings is associated with an additional overhead related to the proof of validity approach, but, nonetheless, is sufficiently fast for a seamless interaction experience between adjacent branches.

Communication with remote branches could be established in two different ways. The first occurs with interaction that is multiple hops away from the target chain, involving message passing through a set of interconnected routed chains. Another is faster but requires preliminary agreements between two decentralized networks and is applicable when communication has to be established with a pair of distant branches. Overall, decision automation between these approaches could be derived based on latency, finality, and throughput capabilities of the intermediary hops.

Finally, this article presents an overview and comparison of the proposed state sharding model and existing multichain solutions based both on independent chain interaction and initially integral systems.

Overall, the tenet of this article is to provide a novel approach for building modern multichain systems based on tree-like architectural structures. It is the intention of this article to spark further research towards efficient state sharding models and methods applicable in the context of blockchain technology.

REFERENCES (TRANSLATED AND TRANSLITERATED)

1. Gad, A. G., et al. (2022). Emerging Trends in Blockchain Technology and Applications: A Review and Outlook. *Journal of King Saud University - Computer and Information Sciences, 34(9),* 6719–6742. https://doi.org/10.1016/j.jksuci.2022.03.007
2. Zheng, Z., et al. (2017). An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends. *IEEE International Congress on Big Data (BigData Congress).* https://doi.org/10.1109/BigDataCongress.2017.85
3. Yaga, D., et al. (2019). Blockchain Technology Overview. *National Institute of Standards and Technology Internal Report.* https://doi.org/10.48550/arXiv.1906.11078
4. Golosova, J., & Romanovs, A. (2018). The Advantages and Disadvantages of the Blockchain Technology. *IEEE 6th Workshop on Advances in Information, Electronic and Electrical Engineering (AIEEE).* https://doi.org/10.1109/AIEEE.2018.8592253.

5. Habib, G. et al. (2022). Blockchain Technology: Benefits, Challenges, Applications, and Integration of Blockchain Technology with Cloud Computing. *Future Internet, 14(11).* https://doi.org/10.3390/fi14110341
6. Taherdoost H. (2023). Smart Contracts in Blockchain Technology: A Critical Review. *Information, 14(2).* https://doi.org/10.3390/info14020117.
7. Revoredo, T. (2023). *Blockchain layers.* Medium. https://tatianarevoredo.medium.com/blockchain-layers-e2240baa649a
8. Whiteboard, L. W. (2022). *What is Blockchain Layer 0, 1, 2, 3 Explained.* Medium. https://medium.com/@learnwithwhiteboard_digest/what-is-blockchain-layer-0-1-2-3-explained-56226d4bb2cd
9. Chatzigiannis, P., Baldimtsi, F., & Chalkias, K. (2022). SoK: Blockchain Light Clients. *Financial Cryptography and Data Security. FC 2022. Lecture Notes in Computer Science, 13411.* https://doi.org/10.1007/978-3-031-18283-9_31
10. Thibault, L. T., Sarry, T., & Hafid, A. S. (2022). Blockchain Scaling using Rollups: A Comprehensive Survey. *IEEE Access.* https://doi.org/10.1109/access.2022.3200051
11. Lavaur, T., Lacan, J., & Chanel, C. P. C. (2022). Enabling Blockchain Services for IoE with Zk-Rollups. *Sensors, 22(17).* https://doi.org/10.3390/s22176493
12. Liu, W., et al. (2024). Distributed and Parallel Blockchain: Towards A Multi-Chain System with Enhanced Security. *IEEE Transactions on Dependable and Secure Computing,* 1–16. https://doi.org/10.1109/tdsc.2024.3417531
13. Sion, S. I., et al. (2024). A Comprehensive Review of Multi-chain Architecture for Blockchain Integration in Organizations. *Business Process Management: Blockchain, Robotic Process Automation, Central and Eastern European, Educators and Industry Forum. BPM 2024. Lecture Notes in Business Information Processing, 527.* https://doi.org/10.1007/978-3-031-70445-1_1
14. Hashim, F., Shuaib, K., & Zaki, N. (2023). Sharding for Scalable Blockchain Networks. *SN Computer Science, 4(2).* https://doi.org/10.1007/s42979-022-01435-z
15. Stone, D. (2021). Trustless, Privacy-Preserving Blockchain Bridges. *arXiv.* https://doi.org/10.48550/arXiv.2102.04660
16. *OKX. What is Wormhole? Powering blockchain's interoperability.* (n. d.). OKX. https://www.okx.com/learn/what-is-wormhole
17. Liu, H., et al. (2021). Merkle Tree: A Fundamental Component of Blockchains. *2021 International Conference on Electronic Information Engineering and Computer Science (EIECS).* https://doi.org/10.1109/EIECS53707.2021.9588047
18. Jing, S., et al. (2021). Review and Investigation of Merkle Tree's Technical Principles and Related Application Fields. *2021 International Conference on Artificial Intelligence, Big Data and Algorithms (CAIBDA).* https://doi.org/10.1109/CAIBDA53561.2021.00026
19. Kudzin, A. et al., (2020). Scaling Ethereum 2.0s Cross-Shard Transactions with Refined Data Structures. *Cryptography, 6(4).* https://doi.org/10.3390/cryptography6040057
20. Petrowski, J. (n. d.). *The Path of a Parachain Block. Polkadot.* URL: https://polkadot.com/blog/the-path-of-a-parachain-block.
21. Abbas, H., Caprolu, M., & Di Pietro, R. (2022). Analysis of Polkadot: Architecture, Internals, and Contradictions. *2022 IEEE International Conference on Blockchain (Blockchain),* 61–70. https://doi.org/10.1109/Blockchain55522.2022.00019
22. Essaid, M., Kim, J., & Ju, H. (2023). Inter-Blockchain Communication Message Relay Time Measurement and Analysis in Cosmos. *Applied Sciences, 13(20).* https://doi.org/10.3390/app132011135
23. Peelam, M. S., Chaurasia, B. K., Sharma, A. K., Chamola, V., & Sikdar, B. (2024). Unlocking the Potential of Interconnected Blockchains: A Comprehensive Study of Cosmos Blockchain Interoperability. *IEEE Access, 12,* 171753–171776. https://doi.org/10.1109/ACCESS.2024.3497298
24. *Primary Network | Avalanche Docs. Avalanche Docs.* (n. d.). https://docs.avax.network/protocol/primary-network.
25. *Avalanche L1s | Avalanche Docs. Avalanche Docs.* (n. d.). https://docs.avax.network/protocol/avalanche-l1s

КІБЕРБЕЗПЕКА: освіта, наука, техніка
CYBERSECURITY: EDUCATION, SCIENCE, TECHNIQUE
№ 2 (26), 2024
ISSN 2663 - 4023

Котов Максим Сергійович
магістр кібербезпеки, аспірант кафедри кібербезпеки та захисту інформації
Київський національний університет імені Тараса Шевченка, Київ, Україна
ORCID ID: 0000-0003-1153-3198
maksym_kotov@ukr.net

ДЕРЕВОПОДІБНИЙ ШАРДИНГ СТАНУ ДЛЯ МАСШТАБУВАННЯ ТА БАЛАНСУВАННЯ НАВАНТАЖЕННЯ В МУЛЬТИЧЕЙН СИСТЕМАХ

Анотація. Розвиток технологій, що надають фундамент для підтримки стрімкого зростання попиту на розподілені системи досягнення консенсусу, став однією з глобальних тенденцій у галузі науки та техніки. Технологія блокчейн, її протоколи консенсусу, методи зв'язку та архітектурні підходи, поширені в транзакційних системах нульової довіри. У даному контексті, однією з ключових перешкод, з якою стикаються інженери та учасники взаємодії, є обмежені можливості масштабування, пов'язані з цими системами через вимоги до узгодженості та надійності. Сучасні блокчейн-системи створюють складнощі, пов'язані з управлінням простором зберігання історії, затримками виконання транзакцій і, загалом, пропускною здатністю операцій, що гальмує широку інтеграцію децентралізованих систем у повсякденну діяльність. Щоб обійти ці обмеження, наукові та інженерні спільноти активно розробляють безліч протоколів міжланцюгового зв'язку, стратегій шардингу та стратегій розширення обчислювальних потужностей, щоб пом'якшити початкові логічні обмеження запропонованої технології консенсусу. Тим не менш, розроблені стратегії пов'язані з власними обмеженнями, часто стикаються з центральним вузьким місцем в підході до балансування навантаження або суттєво жертвують властивостями остаточності та затримки. Метою цієї статті є представити та описати підхід шардингу на основі деревоподібних мультичейн систем. По-перше, цей документ описує загальну архітектуру запропонованої мережі, встановлюючи основу, на якій відбувається подальше обговорення. По-друге, набір методів зв'язку за участю батьків, суміжних і віддалених гілок для обміну даними транзакцій. По-третє, запропонована архітектура шардингу та її властивості порівнюються з набором існуючих стратегій для досягнення масштабованості в межах технології блокчейн. Загалом, ця стаття представляє новий підхід до побудови надійних, масштабованих і високоефективних мультичейн систем через структуровану деревоподібну ієрархію взаємодіючих блокчейн-мереж.

Ключові слова: блокчейн; мультичейн системи; шардинг стану блокчейну; балансування навантаження блокчейну; комунікаційні протоколи мультичейн; узгодженість і консенсус в мультичейн; деревоподібна мультичейн система.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Gad, A. G., et al. (2022). Emerging Trends in Blockchain Technology and Applications: A Review and Outlook. *Journal of King Saud University - Computer and Information Sciences, 34(9),* 6719–6742. https://doi.org/10.1016/j.jksuci.2022.03.007
2. Zheng, Z., et al. (2017). An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends. *IEEE International Congress on Big Data (BigData Congress).* https://doi.org/10.1109/BigDataCongress.2017.85
3. Yaga, D., et al. (2019). Blockchain Technology Overview. *National Institute of Standards and Technology Internal Report.* https://doi.org/10.48550/arXiv.1906.11078
4. Golosova, J., & Romanovs, A. (2018). The Advantages and Disadvantages of the Blockchain Technology. *IEEE 6th Workshop on Advances in Information, Electronic and Electrical Engineering (AIEEE).* https://doi.org/10.1109/AIEEE.2018.8592253.
5. Habib, G. et al. (2022). Blockchain Technology: Benefits, Challenges, Applications, and Integration of Blockchain Technology with Cloud Computing. *Future Internet, 14(11).* https://doi.org/10.3390/fi14110341

6. Taherdoost H. (2023). Smart Contracts in Blockchain Technology: A Critical Review. *Information, 14(2).* https://doi.org/10.3390/info14020117.
7. Revoredo, T. (2023). *Blockchain layers.* Medium. https://tatianarevoredo.medium.com/blockchain-layers-e2240baa649a
8. Whiteboard, L. W. (2022). *What is Blockchain Layer 0, 1, 2, 3 Explained.* Medium. https://medium.com/@learnwithwhiteboard_digest/what-is-blockchain-layer-0-1-2-3-explained-56226d4bb2cd
9. Chatzigiannis, P., Baldimtsi, F., & Chalkias, K. (2022). SoK: Blockchain Light Clients. *Financial Cryptography and Data Security. FC 2022. Lecture Notes in Computer Science, 13411.* https://doi.org/10.1007/978-3-031-18283-9_31
10. Thibault, L. T., Sarry, T., & Hafid, A. S. (2022). Blockchain Scaling using Rollups: A Comprehensive Survey. *IEEE Access.* https://doi.org/10.1109/access.2022.3200051
11. Lavaur, T., Lacan, J., & Chanel, C. P. C. (2022). Enabling Blockchain Services for IoE with Zk-Rollups. *Sensors, 22(17).* https://doi.org/10.3390/s22176493
12. Liu, W., et al. (2024). Distributed and Parallel Blockchain: Towards A Multi-Chain System with Enhanced Security. *IEEE Transactions on Dependable and Secure Computing,* 1–16. https://doi.org/10.1109/tdsc.2024.3417531
13. Sion, S. I., et al. (2024). A Comprehensive Review of Multi-chain Architecture for Blockchain Integration in Organizations. *Business Process Management: Blockchain, Robotic Process Automation, Central and Eastern European, Educators and Industry Forum. BPM 2024. Lecture Notes in Business Information Processing, 527.* https://doi.org/10.1007/978-3-031-70445-1_1
14. Hashim, F., Shuaib, K., & Zaki, N. (2023). Sharding for Scalable Blockchain Networks. *SN Computer Science, 4(2).* https://doi.org/10.1007/s42979-022-01435-z
15. Stone, D. (2021). Trustless, Privacy-Preserving Blockchain Bridges. *arXiv.* https://doi.org/10.48550/arXiv.2102.04660
16. *OKX. What is Wormhole? Powering blockchain's interoperability.* (n. d.). OKX. https://www.okx.com/learn/what-is-wormhole
17. Liu, H., et al. (2021). Merkle Tree: A Fundamental Component of Blockchains. *2021 International Conference on Electronic Information Engineering and Computer Science (EIECS).* https://doi.org/10.1109/EIECS53707.2021.9588047
18. Jing, S., et al. (2021). Review and Investigation of Merkle Tree's Technical Principles and Related Application Fields. *2021 International Conference on Artificial Intelligence, Big Data and Algorithms (CAIBDA).* https://doi.org/10.1109/CAIBDA53561.2021.00026
19. Kudzin, A. et al., (2020). Scaling Ethereum 2.0s Cross-Shard Transactions with Refined Data Structures. *Cryptography, 6(4).* https://doi.org/10.3390/cryptography6040057
20. Petrowski, J. (n. d.). *The Path of a Parachain Block. Polkadot.* URL: https://polkadot.com/blog/the-path-of-a-parachain-block.
21. Abbas, H., Caprolu, M., & Di Pietro, R. (2022). Analysis of Polkadot: Architecture, Internals, and Contradictions. *2022 IEEE International Conference on Blockchain (Blockchain),* 61–70. https://doi.org/10.1109/Blockchain55522.2022.00019
22. Essaid, M., Kim, J., & Ju, H. (2023). Inter-Blockchain Communication Message Relay Time Measurement and Analysis in Cosmos. *Applied Sciences, 13(20).* https://doi.org/10.3390/app132011135
23. Peelam, M. S., Chaurasia, B. K., Sharma, A. K., Chamola, V., & Sikdar, B. (2024). Unlocking the Potential of Interconnected Blockchains: A Comprehensive Study of Cosmos Blockchain Interoperability. *IEEE Access, 12,* 171753–171776. https://doi.org/10.1109/ACCESS.2024.3497298
24. *Primary Network | Avalanche Docs. Avalanche Docs.* (n. d.). https://docs.avax.network/protocol/primary-network.
25. *Avalanche L1s | Avalanche Docs. Avalanche Docs.* (n. d.). https://docs.avax.network/protocol/avalanche-l1s

(cc) BY-NC-SA

This work is licensed under Creative Commons Attribution-noncommercial-sharealike 4.0 International License.